Commun. Comput. Phys.
doi: 10.4208/cicp.080213.151113a

Vol. **16**, No. 1, pp. 115-135
July 2014

Acceleration Strategies Based on an Improved Bubble Packing Method

Nan Qi, Yufeng Nie* and Weiwei Zhang

Department of Applied Mathematics, School of Science, Northwestern Polytechnical University, 127 Youyi West Road, Xi'an 710129, China.

Received 8 February 2013; Accepted (in revised version) 15 November 2013

Available online 10 April 2014

Abstract. The bubble packing method can generate high-quality node sets in simple and complex domains. However, its efficiency remains to be improved. This study is a part of an ongoing effort to introduce several acceleration schemes to reduce the cost of simulation. Firstly, allow the viscosity coefficient c in the bubble governing equations to change according the coordinate of the bubble which are defined separately as odd and normal bubbles,and meanwhile with the saw-shape relationship with time or iterations. Then, in order to relieve the over crowded initial bubble placement, two coefficients w_1 and w_2 are introduced to modify the insertion criterion. The range of those two coefficients are discussed to be $w_1=1$, $w_2\in[0.5,0.8]$. Finally, a self-adaptive termination condition is logically set when the stable system equilibrium is achieved. Numerical examples illustrate that the computing cost can significantly decrease by roughly 80% via adopting various combination of proper schemes (except the uniform placement example), and the average qualities of corresponding Delaunay triangulation substantially exceed 0.9. It shows that those strategies are efficient and can generate a node set with high quality.

AMS subject classifications: 65L50, 65B99

Key words: Bubble packing method, algorithm efficiency, viscosity coefficient, mesh generation, Delaunay triangulation.

1 Introduction

Engineering analysis of mechanical systems have been addressed by deriving a partial differential equation systems through basic physical principles such as equilibrium, conservation of energy and mass, the laws of thermodynamics and Newton's laws of motion. However, once formulated, solving the resulting mathematical models is often intractable, or even impossible, especially when the resulting models are non-linear. Over

*Corresponding author. *Email addresses:* `appleflowerqn@163.com` (N. Qi), `yfnie@nwpu.edu.cn` (Y. Nie), `zhangweiwei@mail.nwpu.edu.cn` (W. Zhang) the past few decades many such problems have emerged in various computer applications, and along with a corresponding collection of solution techniques. These applications include engineering analysis, computer graphics, layered manufacturing, surface design, and shape reconstruction [1].

In recent dozens of years, the finite element method (FEM) is the dominant discretization technique in solving such mechanical problems, while the basic concept in the physical interpretation of the FEM is the subdivision of the mathematical model into disjoint (non-overlapping) components of simple finite elements. For those numerical methods fall into the mesh-based category, the accuracy and convergence properties are dependent on the size and shape of the elements, which consequently have a positive influence on the distribution of the corresponding nodes set [2].

In this paper, attentions are fixed on analysing the node placement method called Bubble Packing Method (BPM), or bubble meshing developed by Shimada et al. [1, 3–5], which is on account of the fact that a pattern of closely packed circles mimics a Voronoi diagram so that a set of well-shaped Delaunay triangles can be created by connecting the centers of the circles, this packed configuration is obtained by defining proximity-based interacting forces among the circles and finding a force-balancing configuration using dynamic simulation.

Since then, a fully automated quadrilateral meshing for finite element analysis was presented, in which the directionality of the mesh is precisely controlled [6–8]. Itoh and Shimada [9] presented a triangular-to-quadrilateral mesh conversion method that can control the directionality of the output quadrilateral mesh according to a user-specified vector field. Additionally, Yamakawa et al. [10] suggested an algorithm for anisotropic tetrahedral meshing.

Kim et al. (2003) [11] employed a modified octree technique to place initial bubbles in three-dimensional bodies, which leads to a drastic reduction of errors for each of the two- and three-dimensional domains, from the present bubble packing technique combined with the adaptive refinement based upon Zienkiewicz and Zhu error estimator [12, 13]. Chung et al. [14, 15] then proposed a new remeshing algorithm using the BPM in two-dimensional finite element analysis and applied this algorithm to the large deformation problem, it works well at the region with large error, it is further able to control the refinement area and the new mesh size easily through the maximum permissible relative error $\eta^*_{\max}$ and the bubble size control factor q. Nie et al. [2] verified the bubble system convergence by studying the changes of the average speed in the dynamic simulation, and also pointed out that physical parameters (e.g., damping coefficient) should be properly selected, which directly impact on the convergence speed of the bubble system. Moreover, parallelization for large-scale problems can also be easily proposed for reduction of the computational cost [16]. Rossi and Shimada et al. [17–19] presented a computerized planning scheme for prostate cryosurgery using a variable insertion depth strategy based on BPM in comparison with the experimentally applied cryoprobe layout, and it is commented that this method was associated with a high computational cost.

Recently, Wu et al. [20] employed BPM into simulating both square and polar liddriven cavity flows under different Reynolds numbers. Cai et al. [21] proposed a numerical path integration method based on bubble grids for nonlinear dynamical systems-Duffing oscillator and Duffing Rayleigh oscillator subjected to harmonic and stochastic excitations. The meshing problem has thus been mostly solved, for different shapes of mesh, or even for meshless method [22]. The advantages of BPM are widely acceptable-consistent applicability to 1D/2D/surface/3D/hybrid domains, precise node spacing control, well-shaped elements, adaptive remeshing capability, intuitive and easy to implement. Nevertheless, like all other physically-based approaches, the second stage of the bubble method, the dynamic simulation is computationally costly due to the pair-wise interbubble force interaction [1], in this sense, a new challenge now is to enable this algorithm efficient rather than taking so much time.

Several modifications have been done to reduce the cost of the simulation previously in [23]. Firstly, allow the viscosity coefficient c gradually increases instead of being taken as a constant, which helps to speed up the convergence. Moreover, at the end of each round simulation, in which bubble additions or deletions are operated, c is assigned to be a relatively smaller value in order to ensure the quality of bubble distribution. Secondly, as solving the ordinary differential equations (ODEs) that control the movement of bubbles, a second-order Euler algorithm instead of a fourth-order Runge-Kutta algorithm is chosen. Finally, the sort process of overlapping rate of bubbles is removed, which is replaced by only setting threshold for bubbles additions and deletions. It is showed that the computing cost decreases by approximately 40% and still can generate a nodes set with high-quality. This modified node placement method is referred as *a Fast Node Placement Method with Bubble Simulation (FNPBS)*.

This paper is ongoingly aiming at improving the BPM in three ways: by defining coefficients w_1 and w_2 to control the density of initial bubble configuration, by setting a threshold to terminate the simulation process, and also by letting the viscosity coefficient c shift with the position of bubble, or precisely, with the determinant of whether its adjacent bubbles being operated with the insertions or deletions, which in turns works upon the governing ODEs model. Four examples, representing four types of domains or node configurations, are executed with the above strategies, which may prove to substantially reduce the computational cost, and likewise expediting the rate of convergence. It is also shown that those strategies are selectively effective to different cases associated with the shape of the domain and the defined ideal distance function.

2 Methodology

The physically-based model for mesh generation, presented as a ”dynamic bubble system” was originally inspired by the idea of bubble meshing and the technology of molecular dynamics. It treats the computational domain to be a force region with viscosity defined on it, while each node in this domain to be the center of a bubble which are drove by the repelling and attracting interbubble forces based on the proximity. The motion of every bubble satisfies the Newton's 2nd Law. During the simulation, insertion and deletion operations of bubbles are implemented according to the bubble's overlapping ratio. As the iterative process goes forward, a closely force-equilibrium configuration of nodes is obtained, which is indeed the node distribution satisfying the given node spacing function. In addition, the mean velocity of bubbles in this system tends to be zero.

While the principle of bubble packing are well documented in the literature [1,22], the method is presented here in brief-for the completeness of presentation-with emphasis on the new modifications.

2.1 Analysis of time consumption

The basic idea of BPM can be primarily summarized into 5 steps:

(A) Initialization

  (a) Initial bubble placement,

  (b) Build the initial adjacent list for each bubble.

(B) Searching and updating of adjacent list.

(C) Calculation of the interbubble force.

(D) Calculating the new position of each bubbles by the governing equations.

(E) Bubble population control with insertion and deletion operations.

Amongst those, except the first step is solely executed at the beginning, the rest are all executed in several rounds, in which there are a series of simulation.

Example 3 in Section 3.3 is used throughout Section 2 for demonstration. It is a $[-3,3]\times[-3,3]$ square region, and the node spacing function is

$$d(x,y)=0.2\times\left|\sqrt{x^2+y^2}-2.0\right|+0.15.$$

The time consumed by adopting the traditional BPM to obtain the node distribution of this region is given in Table 1. All the examples in this paper, if not mentioned specifically, are executed on Intel(R) Core TM2 Duo CPU T6600 2.20GHz, Microsoft Windows XP Professional SP3, complied by Microsoft Visual C++ 6.0(SP6).

Table 1: Time consumed in each sub-step by the traditional BPM.

	time consumption/s	percentage/%
initialization	0.328	0.109
adjacent list search and update	15.123	5.028
calculating forces	36.062	11.991
calculating the new position	153.956	51.195
bubble population control	95.250	31.674

From Table 1, it is noteworthy that the calculation of bubble new position and bubble population control in BPM occupy more than 80% of the entire simulation, which points out the main focus of this study.

2.2 Changing viscous coefficient c with bubbles' position

The BPM treats the problem of establishing a set of discrete points by bubble movement as a force-balance problem of particle motions in a viscous medium. In view of this, strategies are taken to optimize the design of the coefficients in the bubble motions governing equations so that the particle equilibrium can be quickly obtained, and meanwhile to obtain a high-quality nodal distribution.

The bubble system is regarded as a viscous medium system consisting a large amount of particles, in which each particle is dragged by two types of forces, i.e., the interaction force from other particles and the viscous damping force from the virtual medium. The expression of interbubble force, or the intermolecular Van der Waals force in nature is given by [1,16,23]

$$f(\lambda)=\begin{cases} k_0\Big(\dfrac{5}{4}\lambda^3-\dfrac{19}{8}\lambda^2+\dfrac{9}{8}\Big), & 0\le\lambda\ge 1.5,\\ 0, & \lambda>1.5, \end{cases} \tag{2.1}$$

where λ represents the ratio of the actual distance l and the desired distance between two bubbles, and k_0 is a parameter to scale the force value. The general form of this function is shown in Fig. 1.

As explained previously, the final force balancing configuration of bubbles is obtained by solving a system of non-linear ODEs. In the BPM, the dynamic bubble system is represented according to the Newton's 2nd Law:

$$m\frac{d^2x_i(t)}{dt^2}+c\frac{dx_i(t)}{dt}=F_i(t), \tag{2.2}$$

where m denotes mass, c is the damping coefficient, and x_i is the position of the ith bubble's central point, F_i is the resultant force exerted on the ith bubble.

An essential issue in this physically-based approach is the selection of physical parameters such as the mass, the damping and the strength of interbubble forces, which is measured in terms of the coefficient of the interbubble force. They directly determine the time response of the ODEs system and so that highly effect the efficiency and convergence of the bubble system. Therefore, it is intended to design a "well-behaved" physical model by carefully choosing a combination of parameters.

Shimada et al. [1] once pointed out that the system is similar to a standard second-order, single degree system consisting of a mass, viscous damping, and a linear spring. By defining representative linear spring constants for this non-linear springs, it is concluded that physical parameters should be chosen such that the damping ratio ζ gives

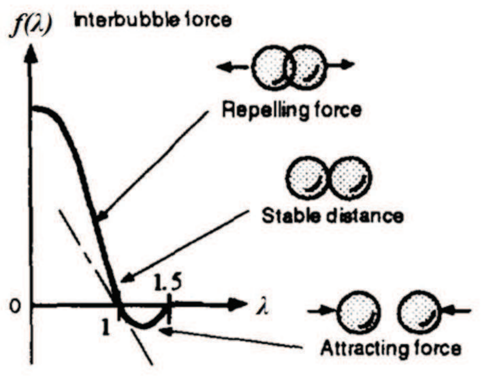


Figure 1: The Van der Waals force, which exerts a repelling force when two bubbles are located closer than a specified distance l_0, and an attracting force when two bubbles are farther away [1].

approximately

$$\zeta = \frac{c}{2\sqrt{mk_0}} \approx 0.7,$$

in which all the parameters are defined as constants.

To study the influence of damping coefficient c on the convergency of the bubble system, let $m = 1.0$, $k_0 = 1.0$. Compared to the interbubble force, if the viscosity of the medium is too large, the particles can quickly tend to the balance, but it is difficult to obtain a high-quality nodes distribution; while if the viscosity is too small, then at the beginning of the simulation, the bubble system is conducive to converge to a suitable nodes set, it is because the smaller the value of c, the larger displacement of bubbles in every round of simulation, so that the better position of bubbles will be acquired at the early stage. However, the defect of this assignment leads to relatively longer time consumed for the whole system to dynamically reach an equilibrium state.

The analysis above is concluded by taking the viscous coefficient c as a constant. However, the node placement process is not the movement of particles in the viscous liquid in reality. This is simply a simple mathematical model in virtue of the embedded physical meaning, which means the damping coefficient c in the ODEs is not necessarily required to be a constant. If setting c increasing linearly over time (steps), it will help the system accelerate to converge. Due to the operation of deletion and insertion of bubbles at the end of each round of simulation, the coefficient c shall be further adjusted to a relatively small value [23].

In addition, it also intends to design a relation between damping coefficient with the position of the bubble itself. It is because after the deletion or insertion based on the bubble's overlapping ratio, not all the bubbles' position need to be reoptimized to a large extent except the ones close to the deleted or newly inserted bubbles, those are called *odd bubbles*, otherwise *normal bubbles*. In this sense, for the odd bubbles tagged in each round of simulation, their viscous coefficients need to be assigned as a small value c_{odd}, so that the basic incremental sawtooth-shape variance scheme shall be modified to be flat

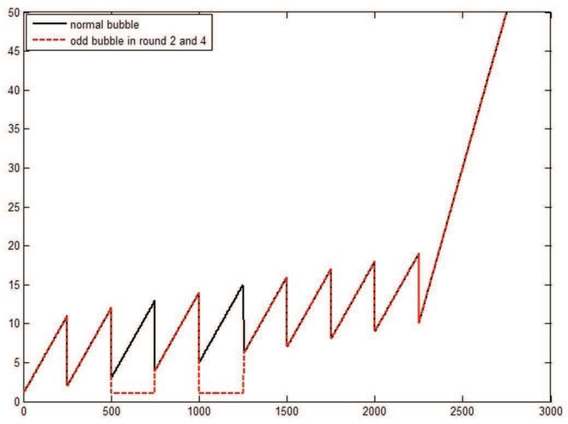


Figure 2: The viscosity coefficient of two representative bubbles as a function of time: one is a normal bubble (black solid line) throughout the whole simulation, the curve turns out to be saw-shaped and roughly increased pattern. Analysed in [23], there are three parameters c_0, k_1, k_2 to determine the shape of this curve, where c_0 is the initial viscous term, k_1, k_2 are the slopes for viscous coefficients amongst inter-round and extra-round simulations, $0<k_1<k_2$ is studied to be satisfied, firstly to construct the term c at the start of this round to be smaller than the end of last round, and secondly to ensure it is a increased incremental pattern; the another is an odd bubble (red dashed line) in the 2nd and 4th rounds, specifically, while in other round it is still a normal bubble so that its viscous changes with other normal bubbles.

when applied to those odd bubbles, while for others, basically linear increased scheme is enough to attain the normal bubbles stably and quickly converge to the final stage. This bifurcated modification allows the bubbles in different situations in each round of simulation to move to superior positions as quickly as possible, and also helps to ensure the quality of bubble distribution. Fig. 2 gives one method to realize the above scheme to set the damping coefficient c changing both over time (steps) and position.

To achieve this scheme, the basic idea is to detect the nodes whose overlapping ratio do not meet the requirements so as to have insertion or deletion operations at the end of each round of simulation. The coordinates of those nodes are recorded since it is the only variable that is kept identical and can be transmitted from this round to next round, the nodes count, however, has generally been arranged in sequential order. Thus, if the bubble is detected as an odd bubble, then the viscous term for next round will be set to be $c_{odd}=1$ here; otherwise, it follows the change with time/iterations.

Since at the first several round of simulation, the bubble population control operation will affect most of the bubbles. In order to save the time for detection, searching and assigning, the c varying with position scheme will be operated when $N_1-N_2\leq 0.05\%\times N_1$, where N_1, N_2 are the nodes count before and after this round of simulation, respectively. At this time, the whole bubble configuration has been formed and the position changes only involve a small number of nodes.

Adopting it into Example 3, the result shows that the scheme actually increases the computing time. This is mainly due to the addition of searching operation, which not only offset the time saved, but cost more.

	time consumed/s	initial node count	final node count
WITHOUT this scheme	187.156	1420	712
WITH this scheme	194.594	1420	709

Nevertheless, if setting the node spacing function to be

$$d(x,y)=0.35\times\left|\sqrt{x^2+y^2}-2.0\right|+0.05,$$

namely enlarge the node spacing variance amongst bubbles. Conducting it on an 8-core mainframe computer, the result in Fig. 3 shows that this scheme does improve the performance of the computing efficiency, but requires that the insertion and deletion operations are confined to some local areas, for instance, the crack propagation problems. It needs to be pointed out that the comparison is solely between the simulation with and without our scheme in the same environment. In other words, the eight-core computations does not contrast with single-core ones. The multicore simulation is due to the increased time consumption when the node spacing being largely reduced.

	time consumption/s	initial node count	final node count
WITHOUT this scheme	19.61	1822	1602
WITH this scheme	9.15	1822	1598

The plots of the 2-D meshes in Fig. 3 shows that the algorithm produces triangles that are almost equilateral. One commonly used mesh quality measure is the ratio between the radius of the largest inscribed circle (times two) and the smallest circumscribed circle [24], which is very similar to the concept of "radius ratio":

$$q=2\frac{r_{in}}{r_{out}}=\frac{(b+c-a)(c+a-b)(a+b-c)}{abc},$$

where a, b, c are the side lengths. An equilateral triangle has $q=1$, while a degenerate triangle (zero area) has $q=0$. The coarse and dense Delaunay meshes have $q=0.9558$ and $q=0.9425$, respectively, which are both desirable properties when solving PDEs with the FEM.

2.3 Reducing the time consumed for bubble population control

It is essential to obtain a advantageous initial bubble configuration before physically-based relaxation since a suitable first guess will greatly reduce the convergence time of the lengthy relaxation process [1]. The bubbles initial placing scheme in the BPM based on a the repeated bisection technique or hierarchical spatial subdivision (HSS) has been devised. This scheme is firstly applied to the boundaries of the region-in 2D is a segment, in 3D is first segment then surface, afterwards, operating the placing for the inner part of

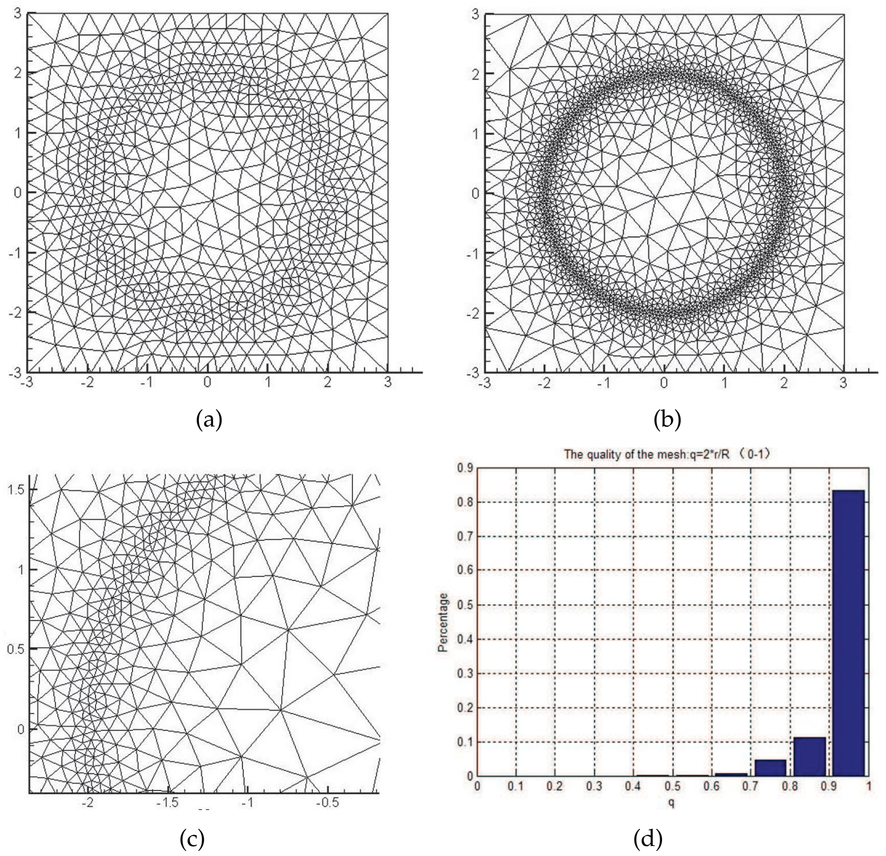


Figure 3: (a) the coarse node configuration, (b) the dense node configuration, (c) the locally zoom-in, (d) the Delaunay radius ratio of the dense node configuration.

the region. The process is shown in Fig. 4. This scheme is simple, relatively fast and has been largely adopted [2–8,10,22]. However, when the mesh spacing varies, it can produce initial bubble configurations with large overlapped bubbles. The HSS method does not effect the quality of the final mesh since the population control algorithm ensures the correct number of final bubbles, it does slow down the meshing processing since a large number of movement steps are required to obtain a force balanced condition.

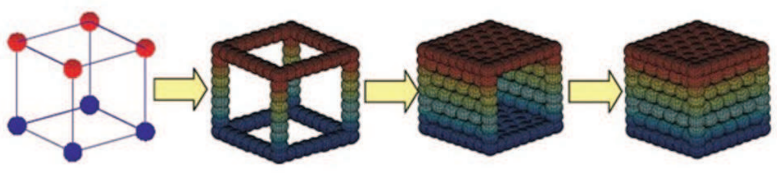


Figure 4: The packing order in a 3D case [2].

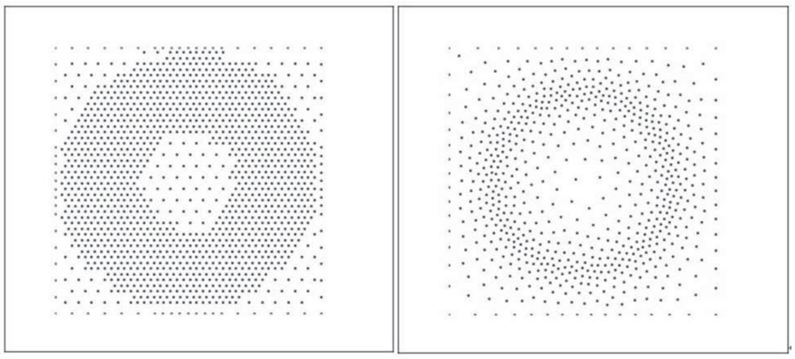


Figure 5: The comparison between the initial 1420 nodes distribution and the final 714 nodes distribution.

Accordingly, Yokoyama et al. [25,26] take a different approach to get the initial configuration, specifically, an initial triangulation of the boundary points is used to define a series of trial points, and each of these trial points is tested in turn to determine the suitability of adding a bubble at that point. This method can easily handle variable mesh spacing without leaving gaps or producing overlapped bubbles. However, it is admitted that the overall technique is comparatively slow compared to the one using the HSS as the initial bubble placing method. In other words, it is worthwhile to improve the HSS method in the aspect of reducing the number of the large overlapped bubbles.

From Fig. 5, it is clear to see that during the 10 rounds simulation, the number of bubbles reduces from initially 1420 to finally 714, which indicates nearly 50% of the initial placed bubbles will be deleted at the end. In this sense, to construct the subdivision somewhat more sparse need to be adapted.

The basic idea of the HSS is to subdivide a curve, a surface, or a volume hierarchically using a binary tree, a quadtree, and a octree respectively. As a result, bubbles are inserted until they cover the entire region without significant gaps or overlaps. Regardless of to place a initial boundary bubble or a initial inner bubble, the criterion for whether to place a bubble between two end points P_1, P_2 is by judging $\overline{P_1P_2} \geq d_1/2+d_2/2$, where d_1, d_2 are the diameters of those end bubbles, which are calculated from the given node spacing function, and $\overline{P_1P_2}$ the distance along the curve segment. If TRUE, meaning that the two end bubbles completely cover the curve segment, and thus no further subdivision is required; If FALSE, then further subdivision is required, and a new bubble is placed at the midpoint of the two end points (Fig. 6). This procedure is recursively repeated until the whole region is covered by bubbles, which is basically the same for surface or volume bubble placement.

To relief the largely coarse initial node placement, two weighting factors w_1, w_2 are considered to modify the criterion to be $w_1 \times \overline{P_1P_2} \geq d_1/2+d_2/2$ for initial boundary nodes, and likewise $w_2 \times \overline{P_1P_2} \geq d_1/2+d_2/2$ for initial inner nodes.

As to the selection of w_1, several experiments are executed by the FNPBS, and the results are given in Table 2.

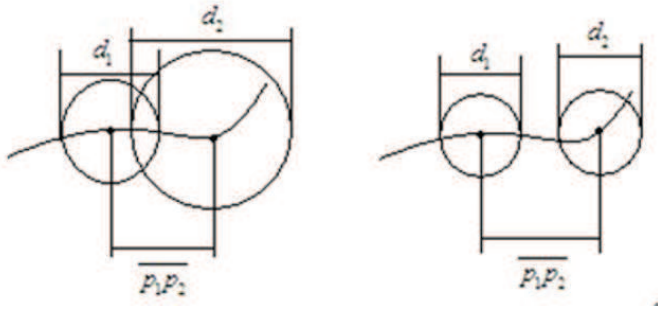


Figure 6: The basic idea of hierarchical spatial subdivision for a curve segment.

Table 2: Results with different w_1.

w_1	initial nodes count	final nodes count	time consumed/s
0.8	1404	780	463.609
0.9	1404	752	421.422
1	**1420**	**712**	**187.156**
1.2	1432	715	192.903
1.5	1460	720	198.813
2	1508	694	278.859

Table 3: Results with different w_2.

w_2	initial nodes count	final nodes count	time consumption/s
1	1420	712	187.156
0.8	1139	711	116.469
0.7	983	707	99.328
0.6	**826**	**707**	**91.515**
0.57	660	655	215.093

Similarly, when going forward to do experiments on w_2, it is obtained with different time consumption shown in Table 3.

Drawing from the form of the modified criterion, $w_1,w_2 \leq 1$ are required. When $w_1=1$, $w_2=0.6$, the radius ratio of the corresponding Delaunay triangulation is 0.9525, with almost no difference with 0.9435 by the FNPBS and 0.9558 by traditional BPM, which means this radius ratio still keeps a rather high level and satisfies the quality requirement for FEM mesh. However, the time consumed in this case is approximately 50% less than the FNPBS, not to mention the traditional BPM.

2.4 Set termination condition

In either the traditional BPM and the FNPBS, the iteration termination condition is to set the round number of bubble packing process, and often this choice is settled based

Table 4: The data for each round of simulation for Example 3 by the FNPBS.

round count	iteration count	time consumed/s	nodes count	Delaunay element count	radius ratio
0	0	0	1420	2758	0.9742
1	250	56.953	1202	2339	0.8562
2	500	93.578	1013	1963	0.8268
3	750	115.625	847	1631	0.8230
4	1000	129.156	748	1434	0.8650
5	1250	138.813	726	1391	0.9052
6	1500	147.438	721	1381	0.9231
7	1750	155.938	716	1371	0.9286
8	2000	164.313	714	1367	0.9351
9	**2250**	**172.719**	**712**	1363	**0.9392**
10	2750	187.156	712	1363	0.9435

on experience. For instance, in the foregoing example, the pre-established round count equals 10, and the simulation time and nodes count for each round are given in Table 4.

As can be seen from the above table, the variance between two successive rounds will be reduced with the increase of iteration. In those last rounds, the nodes variance is very meager, that is to say, at those times, the bubbles' overlapping ratio is basically in line with the requirements. Thus the insertion and deletion operate quite a few so that the corresponding Delaunay radius ratio will not change greatly, and will maintain a relatively stable state, shown in Fig. 7.

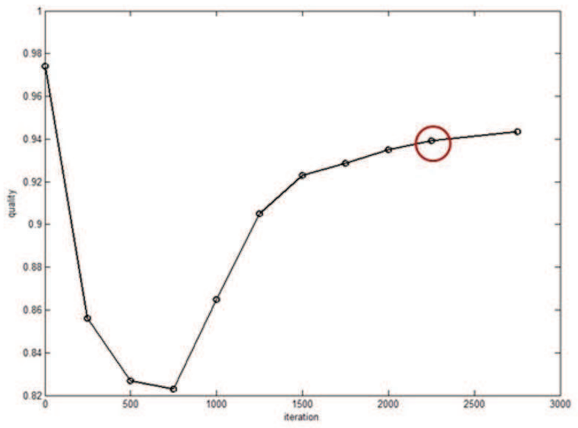
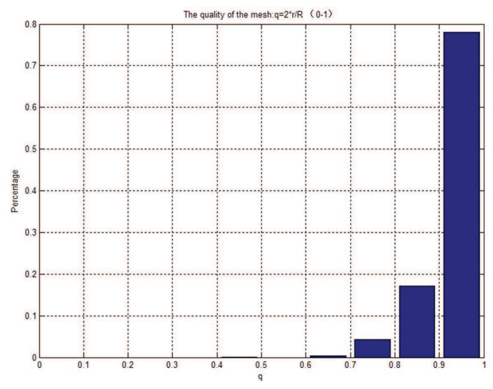
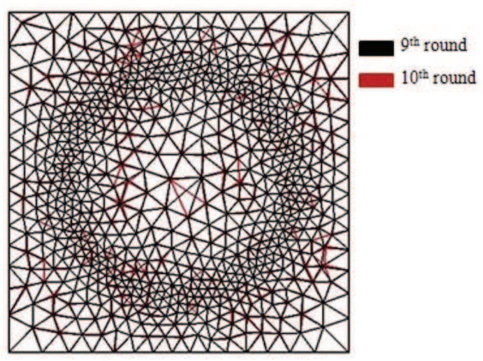


Figure 7: The Delaunay radius ratio changes with the iteration. Note that the oblique quadtree method for 2D initial bubble placement is adopted, which means for roughly uniform node spacing region, the triangulation quality will be rather high, for the node spacing region with great changes, however, the grid shapes will be seriously distorted. Therefore, although the mean radius ratio of the initial node placement seems to be most close to 1, the minimum radius ratio is even less than 0.5, which is quite a poor mesh.



(a)



(b)

Figure 8: (a) The Delaunay radius ratio of 9th round simulation. (b) The triangulation mesh comparison between the 9th and 10th round simulations. There are only some rare differences except some nodes roughly satisfying the rule of empty circumcircle, which have minor influence on the mesh radius ratio.

Drawing in the above discussion and results, the termination condition sets to be when $N_1 - N_2 \leq 0.5\% \times N_1$ is satisfied for twice, where N_1, N_2 are the nodes count before and after this round of simulation, respectively. From Table 4, the end tag lies to 9th round, and the time consumed is 172.719s and the radius ratio is 0.9392 (Fig. 8). If combined with strategy in Section 2.3, the time reduction will be more pronounced, which is shown in the following section.

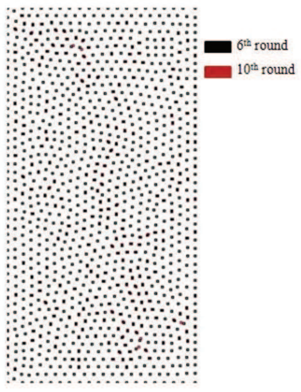
3 Numerical examples

Selectively applying different combination of the above strategies to four types of computing regions, the results of simulation are obtained as follows.

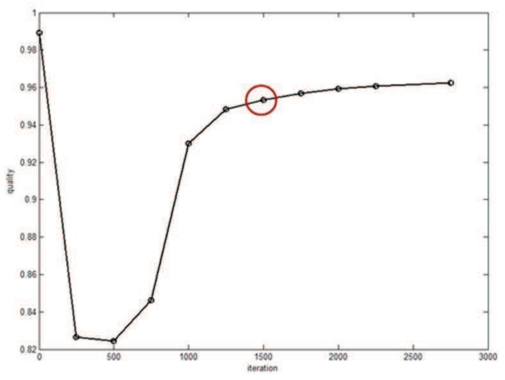
3.1 Uniform distribution on symmetric multiple connected domain

This is a $[-1,1]\times[-2,2]$ rectangular region with node spacing function to be uniformly $d(x,y)=0.1$. w_1 is similarly chosen to be 1, while for w_2, numerical experiments show that when $w_2 \in [0.695,1]$, the node configuration and time consumption are kept the same, when $w_2 < 0.69$, "the debug error" appears. Thus, for this uniform node placement, $w_1 = w_2 = 1$ may be a preferable choice.

It is concluded that using the BPM to uniform node placement is not an optimal selection, even if putting to use several speeding schemes, the time consumption and placement effect are generally less satisfactory compared to some other classic node placement methods, e.g., Advancing Front Triangulation, Sweeping Mesh Generation. In addition, changing the value of w_2 here has no effect either, because if w_2 is chosen to be close to 1, the initial distribution has not changed since the modified criterion treats all the situation



(a)



(b)

Figure 9: (a) The node displacement comparison between the 6th and 10th round simulations. (b) The radius ratio changes with iteration.

the same. However, if w_2 is less than a certain value, here it is approximately 0.69, then a dramatic changes of the initial inner bubbles placement will even cause the number of initial bubbles far less than the ultimately ideal node count, the resulting configuration, therefore, will be highly deformed, or even encounter "debug error".

However, it is undeniable that the third strategy to set the termination condition is still applied here to cause the simulation terminate after reaching a steady state, which will not only achieve a superior quality of node layout, but also greatly reduces the number of iterations and simulation time. Here the simulation is computed to stop in the 6th round (takes 108.406s) rather than the 10th round (takes 154.094s) based on experience. The results are shown in Fig. 9.

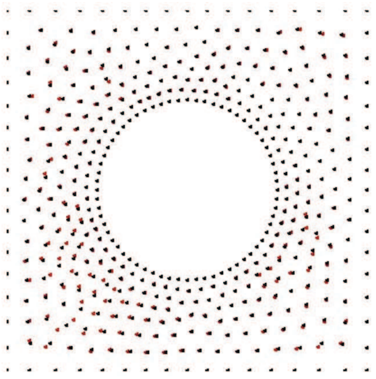
3.2 Nonuniform distribution on symmetric simple connected domain

Here gives a $[-1,1]\times[-1,1]$ square region with a round hole, the node spacing function is

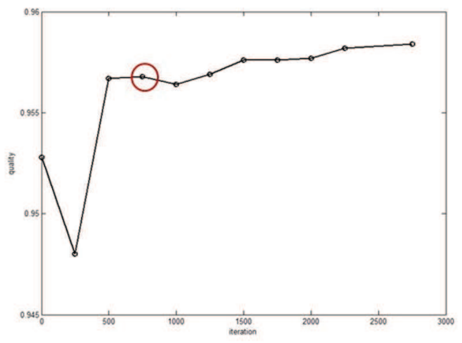
$$d(x,y)=\frac{1}{8}\sqrt{x^2+y^2}.$$

From the FNPBS, after 1250 iterations, the nodes count becomes 587, and the radius ratio is 0.9087.

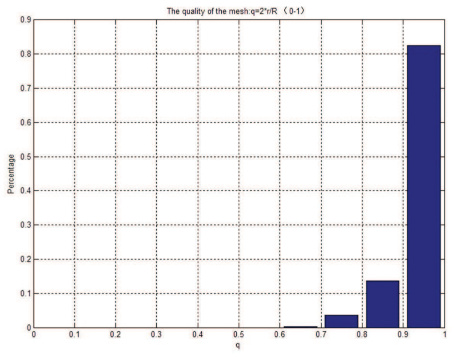
Actually in the 4th round, the nodes count does not come to the specific termination condition and there is still room to improve the overall radius ratio, or the mesh quality. As a result, allow the simulation conduct several more rounds (set to be 10), and since the given number of simulation varies, the functional relationship between viscous term and the number of simulation needs to be altered as well. It turns out that in the 5th round, it meets the condition to stop, and the time consumed is 102.765s, while the radius ratio rises to 0.9396. Furthermore, through a series of trials, $w_1=1$ keeps unchanged, and w_2 is selected to be 0.64. Moreover, the time consumption is only 11s after 4th rounds with



(a)



(b)



(c)

Figure 10: (a) The node displacement comparison between the 4th and 10th round simulations. (b) The radius ratio changes with iteration. (c) The Delaunay radius ratio of 4th round simulation.

radius ratio to be 0.9564, which is a huge time reduction compared with 81.125s by the FNPBS. Some useful results are given in Fig. 10.

3.3 Nonuniform distribution on symmetric multiple connected domain

It is exactly the representative example throughout the explanation in Section 2. To applying together with the strategies in Sections 2.2 and 2.3, and with $w_1=1$, $w_2=0.6$, after computing 3 rounds, the time consumption comes to 29.375s, which is a significant improvement compared to 187.156s by the FNPBS, and the radius ratio is 0.9453, meeting the requirement for the FEM mesh.

3.4 Nonuniform distribution on non-symmetric multiple connected domain

Lastly, a semicircular region shown in Fig. 11(d) with node spacing defined as

$$d(x,y)=\min(\min(h_1,h_2),h_3),$$

Table 5: An overall comparison between the FNPBS and the improved method discussed in this paper.

	time consumed by the FNPBS/s	time consumed with the speeding strategies/s	time reduction percentage	radius ratio by the FNPBS	radius ratio with the speeding strategies
rectangular region	154.094	108.406	**29.65%**	0.9626	0.9532
square region with a hole	82.125	11	**86.61%**	0.9087	0.9564
local dense square region	187.156	29.375	**84.29%**	0.9558	0.9453
semicircular region	117.828	30.266	**74.31%**	0.9407	0.9362

where

$$h_1=0.25-0.15d_1, \qquad h_2=0.05+0.08d_2, \qquad h_3=(d_2-d_1)/2,$$
$$d_1=\sqrt{x^2+y^2}-1, \qquad d_2=\sqrt{(x+0.4)^2+y^2}-0.5,$$

is required to place nodes. By Using strategies mentioned in Sections 2.2 and 2.3, and picking $w_1=1$, $w_2=0.75$, the time consumption reduces to 30.266s after computing 5 rounds. It improves significantly compared to 117.828s by the FNPBS, and the radius ratio is 0.9362, satisfying the requirement for the FEM mesh.

To sum up, Table 5 gives an overall comparison between the FNPBS and the improved method discussed in this paper. From [23], the computing cost decreases by roughly 40% from the traditional BPM to the FNPBS. By further using different strategies here, the cost can further decrease by almost 80% except for the uniform distribution example. All those four examples node placements and mesh results are given in Fig. 11, the node configurations as well as the triangulations are all transitional smooth and uniform.

4 Discussion

4.1 Convergence analysis

Drawing from the bubble system equations of motion, due to the addition of the damping term, the average speed of the bubbles ultimately tends to zero because of the induced viscosity. That is to say, the bubble system eventually converge to a stable equilibrium. The bubble average speed curve with iteration by the traditional BPM, the FNPBS, and the model in this study is shown in Fig. 12, respectively. It is noted that the average initial bubble speed is very large, then rapidly decreases with the increasing number of iteration within one round of simulation. This situation is repeated several times. Moreover, it can

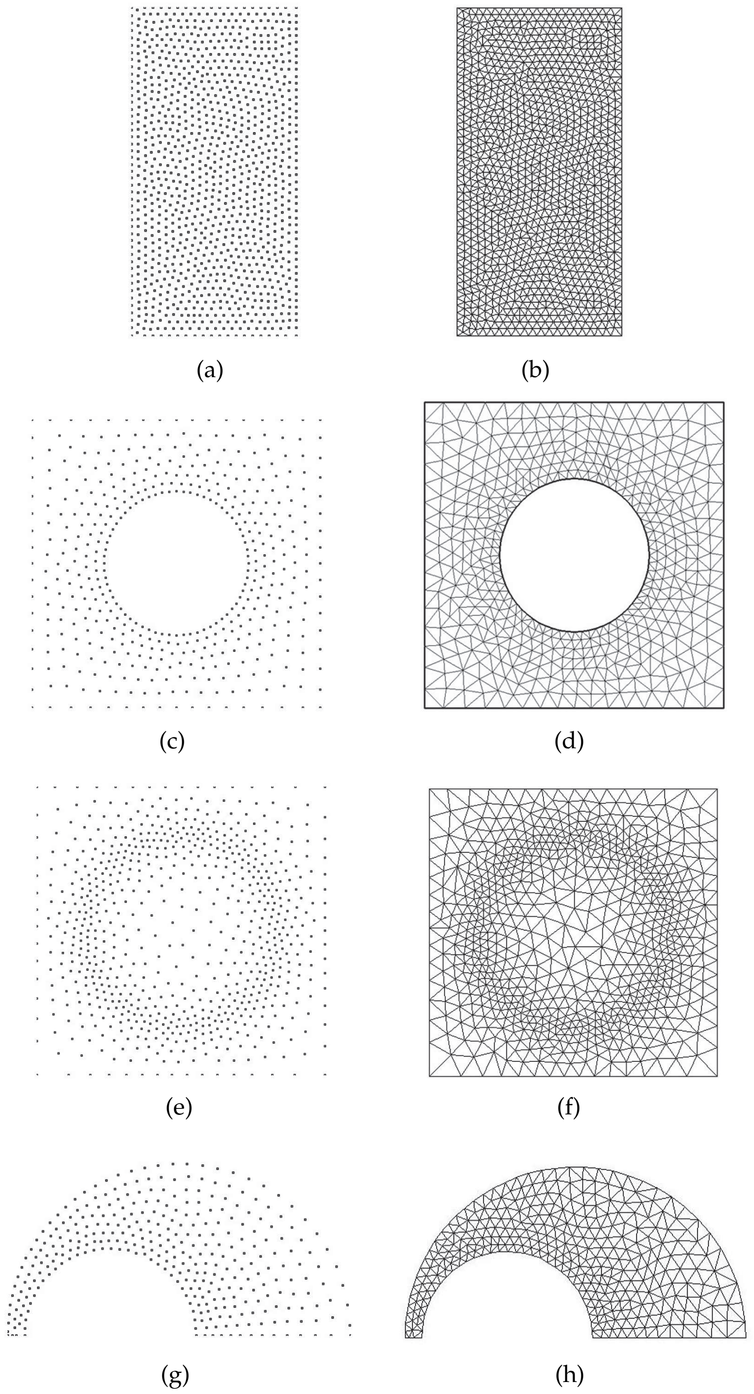


Figure 11: Node placements and corresponding Delaunay mesh configurations to (a), (b) rectangular region; (c), (d)square region with a hole; (e), (f) local dense square region; (g), (h) semicircular region.

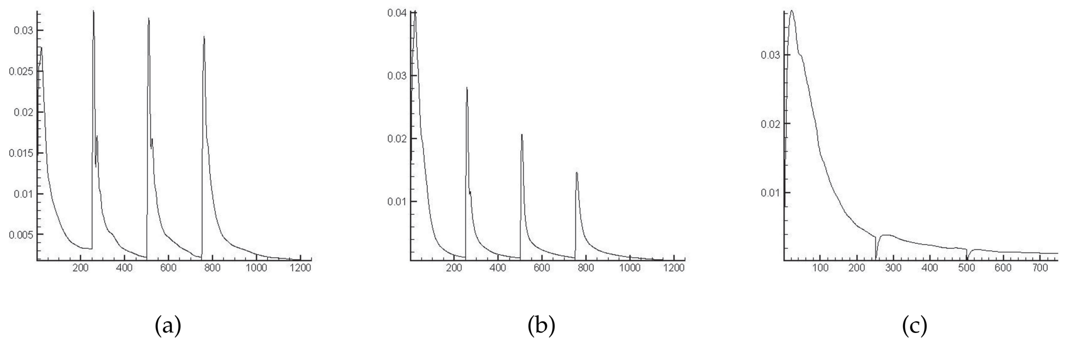


Figure 12: The bubble average speed curve with iteration by (a) the traditional BPM, (b) the FNPBS, and (c) the model in this study.

be explained that in the end of each round, it is necessary to calculate the overlapping ratio of all active bubbles so that the insertions and deletions occur, exactly due to those operations, in the next round, the speed of bubbles becomes relatively large, but will soon decrease.

Comparing those three figures shown in Fig. 12, The fact that the peak speed decreases the fastest in this model indicates that the schemes used herein do help most to force the bubbles to quickly converge to a balanced state. In addition, the impact of the cumulative numerical errors inevitably render the bubble average speed not to be equal to zero but only tend to zero, and the whole system is in dynamic equilibrium.

4.2 The selection of parameters

The parameters in scheme 1 can be summarised as the initial viscous coefficient c_0, the gradient of c within each round k_1, the gradient of c amongst different rounds throughout the simulation k_2 and the modified viscous coefficient of odd bubbles c_{odd}. Generally, $0<k_2<k_1$ since obviously it is linearly increasing pattern, and $k_2<k_1$ ensures the viscous coefficient at the beginning of the current round is smaller than the one at the end of the former round [23]. The c_0 and c_{odd} both have a negligible effect on the time consumption in a relatively large range. Thus, superior results can be obtained as long as the c_0, c_{odd} are not big, say in the range of [1,4].

Clearly from Table 2, the greater the value of w_1, the more of the initial number of nodes, and from what is mostly concerned, the selection to keep w_1 to be unit may be optimal here, which is logical to understand. In each round, bubbles are readjusted to a better configuration by insertion and deletion, so that the boundary nodes can easily run away from the computing region, the projection operation, therefore, is introduced to compel the boundary nodes to move back again dynamically, and generally they are dragged back to their original locations. For this reason, the initial boundary node configuration is actually the final output, and the distances between those nodes should better set to be strictly equal to the given node spacing diameter, which also explains why the value of w_1 slightly bigger than 1 is still acceptable since the criterion is an inequality.

As for the analysis of w_2, in the range of $[0.6,1]$, the time consumed and initial nodes number go into a monotonically decreasing trend. As we can see in Table 3, cutting down the value of weighing factor w_2 indeed generate the initial node placement far less dense, and therefore, the calculation scale and the follow-up computing also attain a corresponding reduction, which turns out to be a much fewer time consumption. However, when w_2 becomes further small, for example below 0.57, the mesh becomes of a deformity or even can not obtain a reasonable result. This is because the selected w_2 is such small that the initial bubbles are not full enough to gain the bubble supplement by insertion operation. Hence, for the demonstration example, $w_2 = 0.6$ appears to be the best choice. Of course, due to various computing region shapes and demands of node spacing, the selection of w_2 should be slightly different, but assigning it to be less than 1 indeed gives rise to greatly help improving the simulation time.

5 Conclusions

By analysing the key factors affect the algorithm efficiency of the traditional BPM, three effective and efficient acceleration schemes are proposed. Example experiments and further discussion show that:

- The scheme by letting viscous coefficient c changes with the position of the bubble apart from with the iteration works most valid when the node spacing varying greatly, such as the FEM mesh used in crack propagation issues.
- By introducing weight factors w_1, w_2 to amend the criterion to control the insertion of initial bubbles, the over crowed initial bubble placement problem can be well solved, while the factor w_1 need to be assigned to be 1 at all the cases, and w_2 is recommended to be within the limits of $[0.5,0.8]$.
- The new termination condition is preferably set to stop the process of the simulation when the bubble system is basically in equilibrium. Simply for this reason, the algorithm transfers to be a self-adaptive method.
- The traditional BPM may not be proper for uniform node placement in regular region, irrespective of employing what scheme to accelerate. Since the BPM is a dynamic system, it is more appropriate for non-uniform configuration. In those cases, the classic mesh generator seems to be more satisfactory both from the aspect in time consumption and the quality of the FE mesh.

In all, the schemes adopted above not only maintain the characteristics of high-quality mesh generation of the original method, but also significantly improve the overall amount of computation and decrease the time consumption, which shows its greatly superior property to the traditional BPM and the FNPBS.

Acknowledgments

This work was sponsored by National Natural Science Foundation of China (No. 11071196, 90916027). We also gratefully acknowledges research support from the Doctorate Foundation of Northwestern Polytechnical University. Large-scale computations were performed using computer facilities in the property of Department of Applied Mathematics in Northwestern Polytechnical University. We appreciate the suggestions from reviewers, which help to improve the paper greatly.

References

[1] K. Shimada, Physically-Based Mesh Generation: Automated Triangulation of Surfaces and Volumes via Bubble Packing, PhD Thesis, Massachusetts Institute of Technology, 1993.

[2] Y. Nie, W. Zhang, Y. Liu and L. Wang, A node placement method with high quality for mesh generation, IOP Conference Series: Materials Science and Engineering, IOP Publishing, 10 (2010), 012–218.

[3] K. Shimada and D. Gossard, Bubble mesh: automated triangular meshing of non-manifold geometry by sphere packing, Proceedings of the Third ACM Symposium on Solid Modeling and Applications, ACM, (1995), 409–419.

[4] K. Shimada and D. Gossard, Automatic triangular mesh generation of trimmed parametric surfaces for finite element analysis, Comput. Aid. Geometric Design, 15(3) (1998), 199–222.

[5] K. Shimada, A. Yamada and T. Itoh, Anisotropic triangulation of parametric surfaces via close packing of ellipsoids, Int. J. Comput. Geometry Appl., 10(04) (2000), 417–440.

[6] K. Shimada, J. Liao and T. Itoh et al., Quadrilateral meshing with directionality control through the packing of square cells, Proceedings of 7th International Meshing Roundtable, (1998), 61–75.

[7] S. Yamakawa and K. Shimada, Quad-layer: layered quadrilateral meshing of narrow two-dimensional domains by bubble packing and chordal axis transformation, Transactions-American Society of Mechanical Engineers Journal of Mechanical Design, 124(3) (2002), 564–573.

[8] S. Yamakawa and K. Shimada, Triangular/quadrilateral remeshing of an arbitrary polygonal surface via packing bubbles, Proc. Geometric Model. Process., 2004 (2004), 153–162.

[9] T. Itoh and K. Shimada, Automatic conversion of triangular meshes into quadrilateral meshes with directionality, Int. J. CAD/CAM, 1(2) (2001), 20–38.

[10] S. Yamakawa and K. Shimada, Anisotropic tetrahedral meshing via bubble packing and advancing front, Int. J. Numer. Methods Eng., 57(13) (2003), 1923–1942.

[11] J. Kim, H. Kim, B. Lee and S. Im, Adaptive mesh generation by bubble packing method, Struct. Eng. Mech., 15(1) (2003), 135–150.

[12] O. Zienkiewicz and J. Zhu, Error estimates and adaptive refinement for plate bending problems, Int. J. Numer. Methods Eng., 28(12) (1989), 2839–2853.

[13] O. Zienkiewicz and J. Zhu, Adaptivity and mesh generation, Int. J. Numer. Methods Eng., 32(4) (1991), 783–810.

[14] S. Chung and S. Kim, A remeshing algorithm based on bubble packing method and its application to large deformation problems, Finite Elements Anal. Design, 39(4) (2003), 301–324.

[15] S. Chung, Y. Choi and S. Kim, Computational simulation of localized damage by finite element remeshing based on bubble packing method, Comput. Model. Eng. Sci., 4(6) (2003), 707–718.

[16] Y. Liu and Y. Nie, Node placement method with bubble simulation and parallelism, Chinese J. Comput. Phys., 6 (2009), 813–820.

[17] M. Rossi, D. Tanaka, K. Shimada and Y. Rabin, Computerized planning of cryosurgery using bubble packing: an experimental validation on a phantom material, Int. J. Heat Mass Transfer, 51(23) (2008), 5671–5678.

[18] D. Tanaka, K. Shimada, M. Rossi and Y. Rabin, Cryosurgery planning using bubble packing in 3d, Comput. Methods Biomech. Biomed. Eng., 11(2) (2008), 113–121.

[19] M. Rossi, D. Tanaka, K. Shimada and Y. Rabin, Computerized planning of prostate cryosurgery using variable cryoprobe insertion depth, Cryobiology, 60(1) (2010), 71–79.

[20] L. Wu, B. Chen and G. Zhou, An improved bubble packing method for unstructured grid generation with application to computational fluid dynamics, Numer. Heat Trans. Part B Fundamentals, 58(5) (2010), 343–369.

[21] L. Cai, Y. Nie, W. Xie and W. Zhang, Numerical path integration method based on bubble grids for nonlinear dynamical systems, Appl. Math. Model., (2013), 1490–1501.

[22] Y. Liu, Y. Nie, W. Zhang and L. Wang, Node placement method by bubble simulation and its application, Comput. Model. Eng. Sci., 55(1) (2010), 89–110.

[23] N. Qi, Y. Nie and W. Zhang, Afast node placement method with bubble simulation, Chinese J. Comput. Phys., 29(3) (2012), 333–339.

[24] P. Persson and G. Strang, A simple mesh generator in matlab, SIAM Review, 46(2) (2004), 329–345.

[25] T. Yokoyama, V. Cingoski, K. Kaneda and H. Yamashita, 3-d automatic mesh generation for fea using dynamic bubble system, Magn. IEEE Transact., 35(3) (1999), 1318–1321.

[26] D. Dibben, 3d mesh generation using bubble meshing for microwave applicators, Magn. IEEE Transact., 36(4) (2000), 1514–1518.